

- 1 -

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	:	Before the Examiner:
Roger Kenneth Abrams	:	Truong, Thanhnga B.
	:	
Serial No.: 09/708,397	:	Group Art Unit: 2135
	:	
Filing Date: November 8, 2000	:	
	:	
Title: SYSTEM AND METHOD	:	IBM Corporation
FOR PREVENTION OF	:	IP Law Dept. YXSA/Bldg. 002
BUFFER OVERFLOW	:	3039 Cornwallis Road
INTRUSIONS	:	P.O. Box 12195
	:	Research Triangle Park, NC 27709

AMENDED SECOND APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I. REAL PARTY IN INTEREST

The real party in interest is International Business Machines Corporation, which is the assignee of the entire right, title and interest in the above-identified patent application.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellant, Appellant's legal representative or assignee which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1-25 are pending in the Application. Claims 1-25 stand rejected. Claims 1-25 are appealed.

IV. STATUS OF AMENDMENTS

Appellant has not submitted any amendments following receipt of the final office action with a mailing date of May 24, 2004.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Independent Claim 1:

In one embodiment of the present invention, a data processing system comprises a bus system. Specification, page 9, lines 8-10; Figure 1, elements 112, 113. The data processing system further comprises a CPU connected to the bus system. Specification, page 9, lines 8-10; Figure 1, elements 110, 112. The data processing system further comprises a RAM connected to the bus system, the RAM being divided into pages, each page having an execution flag. Specification, page 9, lines 10-12; Specification, page 10, lines 10-17; Specification, page 10, line 25 – page 11, line 4; Specification, page 11, lines 14-16; Figure 1, elements 114, 112; Figure 2, elements 202, 206. The data processing system further comprises a memory manager configured to manage the pages of the RAM and permit CPU execution of data on pages according to the execution flag. Specification, page 11, lines 5-17. The data processing system further comprises a program stored within at least one page of the RAM. Specification, page 10, lines 23-25. The data processing system further comprises a program stack stored within at least one page the RAM. Specification, page 11, lines 1-3; Specification, page 12, lines 5-6; Figure 3, element 302. The memory manager is configured to determine whether the program is susceptible to buffer overflow attacks, and, if so, set the execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM. Specification, page 16, lines 8-15.

Independent Claim 9:

In one embodiment of the present invention, a computer program product in a computer-readable medium adapted to prevent buffer overflow attacks comprises a

memory manager code comprising a set of codes operable to direct a data processing system to manage a set of pages within a RAM of the data processing system and to permit a CPU of the data processing system to execute data on pages according to an execution flag on each of the set of pages. Specification, page 11, lines 5-16. The computer program product further comprises an application program code comprising a set of codes operable to direct a data processing system to request the memory manager code to establish a program stack within at least one page the RAM. Specification, page 12, lines 5-10. The computer program product further comprises a susceptibility code comprising a set of codes operable to direct a data processing system to determine whether the application program code is susceptible to buffer overflow attacks, and, if so, set the execution flag for the program stack pages to deny CPU execution of data on the program stack pages. Specification, page 16, lines 8-15.

Independent Claim 18:

In one embodiment of the present invention, a method for handling buffer overflow attacks against an application program running on a data processing system, having a CPU and a RAM divided into pages, the method comprises the step of designating an execution flag for each page of RAM allocated to a stack of the application program. Specification, page 11, lines 14-16; Specification, page 12, lines 5-10. The method further comprises permitting CPU execution of data on pages of RAM according to the execution flag. Specification, page 11, lines 14-15. The method further comprises determining whether the application program is susceptible to buffer overflow attacks. Specification, page 16, lines 8-10; Figure 7, step 706. The method further comprises that if the application program is susceptible to buffer overflow attacks, setting the execution flag as to the pages of RAM allocated to the stack of the application program. Specification, page 16, lines 11-15; Figure 7, steps 708, 716.

VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1-2, 9-12, 15-19 and 24-25 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Yates, Jr. et al. (U.S. Patent No. 6,397,379) (hereinafter "Yates").

B. Claims 3-8, 13-14 and 20-23 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang (U.S. Patent No. 6,477,612).

VII. ARGUMENT

A. Claims 1-2, 9-12, 15-19 and 24-25 are improperly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates.

The Examiner has rejected claims 1-2, 9-12, 15-19 and 24-25 under 35 U.S.C. §103(a) as being unpatentable over Yates. Paper No. 10, page 2. Appellant respectfully traverses these rejections for at least the reasons stated below.

1. The Examiner has not provided any objective evidence for modifying Yates.

A prima facie showing of obviousness requires the Examiner to establish, *inter alia*, that the prior art references teach or suggest, either alone or in combination, all of the limitations of the claimed invention, and the Examiner must provide a motivation or suggestion to combine or modify the prior art reference to make the claimed inventions. M.P.E.P. §2142. The showings must be clear and particular and supported by objective evidence. *In re Lee*, 277 F.3d 1338, 1343, 61 U.S.P.Q.2d 1430, 1433-34 (Fed. Cir. 2002); *In re Kotzab*, 217 F.3d 1365, 1370, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000); *In re Dembiczak*, 50 U.S.P.Q.2d 1614, 1617 (Fed. Cir. 1999). Broad conclusory statements regarding the teaching of multiple references, standing alone, are not evidence. *Id.*

The Examiner's motivation for modifying Yates to have a memory manager configured to determine whether the program is susceptible to buffer overflow

attacks, and, if so, set the execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM, as recited in claim 1 and similarly in claims 9 and 18, is "since the switch is responsive to a first flag value stored in each table entry, and controls the instruction processor to interpret instructions under, alternately, the first or second instruction set as directed by the first flag value of the table entry corresponding to an instruction's memory page (column 36, line 41)." Paper No. 10, page 5. The Examiner's motivation is insufficient to support a *prima facie* case of obviousness for at least the reasons stated below.

The Examiner's motivation is not a motivation as to why one of ordinary skill in the art would modify Yates to include the above-cited claim limitation. The Examiner has not provided any evidence as to why Yates would be modified to determine whether the program is susceptible to buffer overflow attacks. There is no language in Yates that teaches buffer overflow attacks. Further, the Examiner has not explained how the teaching of a switch being responsive to a first flag value stored in each table entry and controls the instruction processor to interpret instructions under in Yates is support for modifying Yates to determine whether the program is susceptible to buffer overflow attacks. Neither has the Examiner explained how the teaching of a second instruction set as directed by the first flag value of the table entry corresponding to an instruction's memory page in Yates is support for modifying Yates to determine whether the program is susceptible to buffer overflow attacks. The cited passage in Yates is not related to determining whether a program is susceptible to buffer overflow attacks. Neither was the cited passage in Yates related to setting an execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM. The Examiner must provide some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify Yates to have a memory manager configured to determine whether the program is susceptible to buffer overflow attacks, and, if so, set the execution flag for program stack pages of

RAM to deny CPU execution of data on the program stack pages of RAM. M.P.E.P. §2142. The Examiner is merely relying upon his own subjective opinion which is insufficient to support a *prima facie* case of obviousness in rejecting claims 1-25. *In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

2. Yates does not teach or suggest the following claim limitations.

a. Yates does not teach or suggest all of the limitations of claims 1, 9 and 18.

Appellant respectfully asserts that Yates does not teach or suggest "a program stack stored within at least one page of the RAM" as recited in claim 1 and similarly in claim 9. The Examiner cites column 2, lines 28-51 of Yates as teaching the above-cited claim limitation. Paper No. 10, page 3. Appellant respectfully traverses and asserts that Yates instead teaches that the memory is divided into pages for management by a virtual memory manager. Yates further teaches that the program is coded in instructions of the first and second instruction sets and uses first and second data storage conventions. Yates further teaches that the switch is responsive to a first flag value stored in each table entry, and controls the instruction processor to interpret instructions under, alternately, the first or second instruction set as directed by the first flag value of the table entry corresponding to an instruction's memory page. Yates further teaches that the transition handler is designed to recognize when program execution has transferred from a page of instructions using the first data storage convention to a page of instructions using the second data storage convention, as indicated by second flag values stored in table entries corresponding to the respective pages, and in response to the recognition, to adjust a data storage configuration of the computer from the first storage convention to the second data storage convention. There is no language in the cited passage in Yates that teaches a program stack stored within a page of memory. Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claims 1 and 9, since the

Examiner is relying upon an incorrect, factual predicate in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

Appellant further asserts that Yates does not teach or suggest "wherein the memory manager is configured to determine whether the program is susceptible to buffer overflow attacks, and, if so, set the execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM" as recited in claim 1 and similarly in claims 9 and 18. The Examiner cites column 10, lines 31-50; column 55, lines 5-12; column 65, line 63 – column 67, line 60; and column 72, lines 9-19 of Yates as teaching the above-cited claim limitation. Paper No. 10, pages 4-5. Appellant respectfully traverses.

Yates instead teaches that a limit detector is operatively interconnected with the register pointer to detect when a range of registers available for collecting profile information is exhausted and a store unit is operatively interconnected with the limit detector of effect storing the profile information from the general registers to the main memory of the computer when exhaustion is detected. Column 10, lines 31-37. Yates further teaches that an alternative tuning method for TAXi_Control.Profile_Timer_Reload_Constant considers buffer overruns. Column 72, lines 8-9. Yates further teaches that when the range of profile collection registers is full, the profile registers are spilled (536 and 548 of FIG. 5a) to a ring buffer in memory. Column 72, lines 10-12. Yates further teaches that the hot spot detector consumes the profile information from this ring buffer. Column 72, lines 12-13. Yates further teaches that if the profiler overruns the hot spot detector and the ring buffer overflows, then the value in TAXi_Control.Profile_Timer_Reload_Constant is increased, to reduce the frequency at which profiling information is collected. Column 72, lines 13-17. Yates further teaches that alternatively, on a buffer overrun, the frequency at which the hot spot detector runs can be increased. Column 72, lines 17-19. While Yates teaches reducing the frequency at which profiling information is collected if the ring buffer overflows, there is no language in the cited passages that teaches determining whether a program is susceptible to buffer overflow attacks.

Neither is there any language in the cited passages that teaches that if the program is susceptible to buffer overflow attacks then set an execution flag for program stack pages of RAM. Neither is there any language in the cited passages that teaches that if the program is susceptible to buffer overflow attacks then set an execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM. Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claims 1, 9 and 18, since the Examiner is relying upon an incorrect, factual predicate in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

Appellant further asserts that Yates does not teach or suggest "a memory manager configured to manage the pages of the RAM and permit CPU execution of data on pages according to the execution flag" as recited in claim 1 and similarly in claims 9 and 18. The Examiner cites column 2, lines 28-51 of Yates as teaching the above-cited claim limitation. Paper No. 10, page 3. Appellant respectfully traverses and asserts that Yates instead teaches that the memory is divided into pages for management by a virtual memory manager. Yates further teaches that the program is coded in instructions of the first and second instruction sets and uses first and second data storage conventions. Yates further teaches that the switch is responsive to a first flag value stored in each table entry, and controls the instruction processor to interpret instructions under, alternately, the first or second instruction set as directed by the first flag value of the table entry corresponding to an instruction's memory page. Yates further teaches that the transition handler is designed to recognize when program execution has transferred from a page of instructions using the first data storage convention to a page of instructions using the second data storage convention, as indicated by second flag values stored in table entries corresponding to the respective pages, and in response to the recognition, to adjust a data storage configuration of the computer from the first storage convention to the second data storage convention. There is no language in the cited passage in Yates that teaches permitting execution of data on pages according to an execution flag. Instead, Yates

teaches adjusting a data storage configuration of the computer from the first storage convention to the second data storage convention in response to second flag values stored in table entries. There is no language in the cited passage of Yates that teaches that adjusting the data storage configurations involves permitting execution of data on pages according to an execution flag. Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claims 1, 9 and 18, since the Examiner is relying upon an incorrect, factual predicate in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

Appellant further asserts that Yates does not teach or suggest "designating an execution flag for each page of RAM allocated to a stack of the application program" as recited in claim 18. The Examiner cites column 2, lines 28-51 of Yates as teaching the above-cited claim limitation. Paper No. 10, page 3. Appellant respectfully traverses. As stated above, Yates instead teaches that the memory is divided into pages for management by a virtual memory manager. Yates further teaches that the program is coded in instructions of the first and second instruction sets and uses first and second data storage conventions. Yates further teaches that the switch is responsive to a first flag value stored in each table entry, and controls the instruction processor to interpret instructions under, alternately, the first or second instruction set as directed by the first flag value of the table entry corresponding to an instruction's memory page. Yates further teaches that the transition handler is designed to recognize when program execution has transferred from a page of instructions using the first data storage convention to a page of instructions using the second data storage convention, as indicated by second flag values stored in table entries corresponding to the respective pages, and in response to the recognition, to adjust a data storage configuration of the computer from the first storage convention to the second data storage convention. There is no language in the cited passage that teaches designating an execution flag for each page of RAM allocated to a stack. Therefore, the Examiner has not presented a *prima facie* case of obviousness in

rejecting claim 18, since the Examiner is relying upon an incorrect, factual predicate in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

- b. Yates does not teach or suggest all of the limitations of claims 2 and 10.

Appellant further asserts that Yates does not teach or suggest "wherein the memory manager and the CPU are configured to deny CPU execution of data by triggering a hardware interrupt" as recited in claim 2 and similarly in claim 10. The Examiner cites column 73, lines 20-24 of Yates as teaching the above-cited claim limitation. Paper No. 10, page 5. Appellant respectfully traverses and asserts that Yates instead teaches that asynchronous x86 transfers of control from hardware interrupts, page faults, breakpoints, single step or any other x86 exception detected in converter or emulator that must be manifest to the x86 virtual machine. There is no language in the cited passage that teaches denying CPU execution of data. Neither is there any language in the cited passage that teaches denying CPU execution of data by triggering a hardware interrupt. Therefore, the Examiner has not presented a *prima facie* case of obviousness in rejecting claims 2 and 10, since the Examiner is relying upon an incorrect, factual predicate in support of the rejection. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1455 (Fed. Cir. 1998).

- c. The Examiner has not cited to any passage in Yates as teaching the limitations of claims 11, 12, 15-17, 24 and 25.

The Examiner makes the blanket statement that claims 11, 12, 15-17, 24 and 25 are rejected under the same rationale as the rejection to claim 1. Paper No. 10, page 5. However, none of these limitations are recited in claim 1 and therefore have not been addressed. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art

reference (or references when combined) that teach or suggest all of the claim limitations in claims 11, 12, 15-17, 24 and 25, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 11, 12, 15-17, 24 and 25. M.P.E.P. §2142.

- d. The Examiner has not cited to any passage in Yates as teaching the limitations of claim 19.

The Examiner makes the blanket statement that claim 19 is rejected under the same rationale as the rejection to claims 1 and 2. Paper No. 10, page 6. However, these limitations are not recited in claims 1 and 2 and therefore have not been addressed. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claim 19, the Examiner has not established a *prima facie* case of obviousness in rejecting claim 19. M.P.E.P. §2142.

- B. Claims 3-8, 13-14 and 20-23 are improperly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

The Examiner has rejected claims 3-8, 13-14 and 20-23 under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang. Paper No. 10, page 6. Appellant respectfully traverses these rejections for at least the reasons stated below.

1. The Examiner has not provided any objective evidence for combining Yates with Wang.

As stated above, a *prima facie* showing of obviousness requires the Examiner to establish, *inter alia*, that the prior art references teach or suggest, either alone or in combination, all of the limitations of the claimed invention, and the Examiner must provide a motivation or suggestion to combine or modify the prior art reference to

make the claimed inventions. M.P.E.P. §2142. The showings must be clear and particular and supported by objective evidence. *In re Lee*, 277 F.3d 1338, 1343, 61 U.S.P.Q.2d 1430, 1433-34 (Fed. Cir. 2002); *In re Kotzab*, 217 F.3d 1365, 1370, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000); *In re Dembiczak*, 50 U.S.P.Q.2d. 1614, 1617 (Fed. Cir. 1999). Broad conclusory statements regarding the teaching of multiple references, standing alone, are not evidence. *Id.*

The Examiner's motivation for modifying Yates to have a memory manager comprise an annotation API that is configured to annotate within a process structure table the susceptibility of the program to buffer overflow attacks, as recited in claim 3 and similarly in claims 4, 5 is "for managing computer system memory (column 1, line 11 of Wang)." Paper No. 10, page 7. Similarly, the Examiner's motivation for modifying Yates to have a memory manager code that includes the process structure table code as an API, as recited in claim 13 and similarly in claim 14 is "for managing computer system memory (column 1, line 11 of Wang)." Paper No. 10, page 7. The Examiner's motivation is insufficient to support a *prima facie* case of obviousness for at least the reasons stated below.

The Examiner's motivation is not a motivation as to why one of ordinary skill in the art would modify Yates to include the above-cited claim limitation. The Examiner has not provided any evidence as to why Yates would be modified to have a memory manager comprise an annotation API that is configured to annotate within a process structure table the susceptibility of the program to buffer overflow attacks. The passage in Wang cited by the Examiner teaches managing computer system memory. The Examiner has not explained how the statement in Wang that says "managing computer system memory" implies that a person of ordinary skill in the art would modify Yates to have a memory manager comprise an annotation API that is configured to annotate within a process structure table the susceptibility of the program to buffer overflow attacks. The cited passage in Wang is not related to annotating within a process structure table the susceptibility of the program to buffer

overflow attacks. Neither is the cited passage in Wang related to an API configured to annotate within a process structure table the susceptibility of the program to buffer overflow attacks. The Examiner must provide some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify Yates to have a memory manager comprise an annotation API that is configured to annotate within a process structure table the susceptibility of the program to buffer overflow attacks as well as to modify Yates to have a memory manager code that includes the process structure table code as an API. M.P.E.P. §2142. The Examiner is merely relying upon his own subjective opinion which is insufficient to support a *prima facie* case of obviousness in rejecting claims 3-5 and 13-14. *In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

2. Claims 20-23 do not recite an API.

The Examiner has rejected claims 20-23 under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang because Yates does not teach an annotation API. Paper No. 10, page 6. However, claims 20-23 do not recite an annotation API. Appellant respectfully asserts that the Examiner has not specifically pointed out the limitation in claims 20-23 not taught by Yates thereby necessitating Wang. Further, the Examiner has not provided any motivation for modifying Yates to incorporate the limitation in claims 20-23 not taught by Yates. In order to establish a *prima facie* case of obviousness, the Examiner must provide a motivation or suggestion, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art to modify the reference to combine reference teachings. M.P.E.P. §2142. Since the Examiner has not provided any motivation for modifying Yates to incorporate the limitation in claims 20-23 not taught by Yates, the Examiner has not presented a *prima facie* case of obviousness in rejecting claims 20-23. M.P.E.P. §2142.

3. The Examiner has not presented a reasonable expectation of success when combining Yates with Wang.

The Examiner must present a reasonable expectation of success in combining Yates with Wang in order to establish a *prima facie* case of obviousness in rejecting claims 3-8, 13-14 and 20-23. *In re Merck & Co., Inc.*, 800 F.2d 1091, 231 U.S.P.Q. 375 (Fed. Cir. 1986); M.P.E.P. §2143.02.

Yates teaches executing instructions for a computer of a first computer architecture on a computer of a second, different computer architecture. Column 1, lines 13-15.

Wang, on the other hand, teaches that one solution to utilizing larger amounts of RAM is referred to as the PSE-36 method. Column 1, lines 48-49. Wang further teaches that with this method, however, applications need to be rewritten and/or the memory partitioned into ramdisks (sections of memory that appear to be disk drives but are actually in RAM) to obtain a performance benefit, which is not always apparent. Column 1, lines 49-53. Wang further teaches that moreover, PSE-36 operates by copying frames of information into thirty-two-bit application addressable space so that the application can access the information, and any copying operation costs processor, cache, bus and memory cycles. Column 1, lines 54-57. Wang further teaches enabling applications to access an increased amount of physical memory via an extension to virtual memory addressing. Column 1, lines 60-63.

The Examiner has not presented any evidence that there would be a reasonable expectation of success in combining Yates, which teaches executing instructions for a computer of a first computer architecture on a computer of a second, different computer architecture, with Wang, which teaches accessing an increased amount of physical memory via an extension to virtual memory addressing. The Examiner has not provided any evidence as to how a method of executing instructions for a computer of a first computer architecture on a computer of a second, different computer architecture would be combined with a method of accessing an increased amount of physical memory via an extension to virtual memory addressing.

Consequently, the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 3-8, 13-14 and 20-23. M.P.E.P. §2143.02.

4. Claims 3-5 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

Appellant respectfully asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "a process structure in data communication the memory manager" as recited in claim 3 and similarly in claims 4 and 5. The Examiner has not cited to any passage in either Yang or Wang as teaching the above-cited claim limitation. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified any passage in either Yang or Wang that teaches or suggests the above-cited claim limitation, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 3-5. M.P.E.P. §2142.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "wherein the memory manager comprises an annotation API, wherein the annotation API is configured to annotate within the process structure table the susceptibility of the program to buffer overflow attacks, wherein the memory manager is configured to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table" as recited in claim 3 and similarly in claim 4. The Examiner cites column 1, line 63 – column 2, line 8 and column 6, lines 8-22 of Wang as teaching the aspect of an API. Paper No. 10, page 6. The Examiner has not addressed the other limitations in claim 3, as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references

when combined) that teach or suggest all of the claim limitations in claims 3 and 4, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 3 and 4. M.P.E.P. §2142.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "an annotation program in data communication with the process structure table, wherein the annotation program is configured to annotate within the process structure table the susceptibility of the program to buffer overflow attacks, and wherein the memory manager is configured to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table" as recited in claim 5. The Examiner cites column 1, line 63 – column 2, line 8 and column 6, lines 8-22 of Wang as teaching the aspect of an API. Paper No. 10, page 6. The Examiner has not addressed the other limitations in claim 5, as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claim 5, the Examiner has not established a *prima facie* case of obviousness in rejecting claim 5. M.P.E.P. §2142.

5. Claims 6-8 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "wherein the program is configured to call the annotation API if the program is susceptible to buffer overflow attacks, the memory manager is configured to determine susceptibility upon a request to allocate an additional page of RAM for the program" as recited in claim 6 and similarly in claims 7 and 8. The Examiner has not addressed the limitation as recited above. The Examiner simply states that claims 6-8 have limitations similar to those of claims 1 and 3 and thus are

rejected under the same rationale as the rejection to claims 1 and 3. Paper No. 10, page 7. However, claims 1 and 3 do not contain the limitation as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claims 6-8, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 6-8. M.P.E.P. §2142.

6. Claims 13 and 14 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "wherein the memory manager code comprises the process structure table code as an API" as recited in claim 13 and similarly in claim 14. The Examiner has not addressed the limitation as recited above. The Examiner simply states that claims 13 and 14 have limitations similar to those of claim 3 and thus are rejected under the same rationale as the rejection to claim 3. Paper No. 10, page 7. However, claim 3 does not contain the limitation as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claims 13-14, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 13-14. M.P.E.P. §2142.

7. Claims 20 and 21 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "establishing a process structure table; maintaining the process structure table by annotating within the process structure table the susceptibility of the application program to buffer overflow attacks, wherein the step of determining susceptibility to buffer overflow attacks is made with reference to the process structure table" as recited in claim 20 and similarly in claim 21. The Examiner has not addressed the limitation as recited above. The Examiner simply states that claims 20 and 21 have limitations similar to those of claim 3 and thus are rejected under the same rationale as the rejection to claim 3. Paper No. 10, page 7. However, claim 3 does not contain the limitation as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claims 20-21, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 20-21. M.P.E.P. §2142.

8. Claims 22 and 23 are not properly rejected under 35 U.S.C. §103(a) as being unpatentable over Yates in view of Wang.

Appellant further asserts that Yang and Wang, taken singly or in combination, do not teach or suggest "wherein the step of maintaining the process structure table is done once at the beginning of execution of the application program, and the step of determining susceptibility is performed upon each receipt of a request to allocate an additional page of RAM for the application program code" as recited in claim 22 and similarly in claim 23. The Examiner has not addressed the limitation as recited above. The Examiner simply states that claims 22 and 23 have limitations similar to those of claims 3 and 9 and thus are rejected under the same rationale as the rejection to claims 3 and 9. Paper No. 10, page 7. However, claims 3 and 9 do not contain the

limitations as recited above. The Examiner bears the burden of establishing a *prima facie* case of obviousness. M.P.E.P. §2142. The Examiner's burden includes identifying a prior art reference (or references when combined) that teach or suggest all of the claim limitations. M.P.E.P. §2142. Since the Examiner has not identified a prior art reference (or references when combined) that teach or suggest all of the claim limitations in claims 22-23, the Examiner has not established a *prima facie* case of obviousness in rejecting claims 22-23. M.P.E.P. §2142.

VIII. CONCLUSION

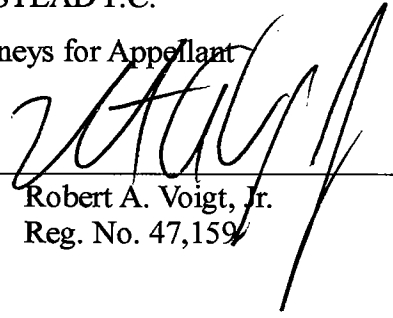
For the reasons noted above, the rejections of claims 1-25 are in error. Appellant respectfully requests reversal of the rejections and allowance of claims 1-25.

Respectfully submitted,

WINSTEAD P.C.

Attorneys for Appellant

By: _____


Robert A. Voigt, Jr.
Reg. No. 47,159

P.O. Box 50784
Dallas, Texas 75201
(512) 370-2832

CLAIMS APPENDIX

1. A data processing system comprising:
 - a bus system;
 - a CPU connected to the bus system;
 - a RAM connected to the bus system, the RAM being divided into pages, each page having an execution flag;
 - a memory manager configured to manage the pages of the RAM and permit CPU execution of data on pages according to the execution flag;
 - a program stored within at least one page of the RAM; and
 - a program stack stored within at least one page the RAM,wherein the memory manager is configured to determine whether the program is susceptible to buffer overflow attacks, and, if so, set the execution flag for program stack pages of RAM to deny CPU execution of data on the program stack pages of RAM.
2. The data processing system of claim 1 wherein the memory manager and the CPU are configured to deny CPU execution of data by triggering a hardware interrupt.
3. The data processing system of claim 1 further comprising:
 - a process structure table in data communication with the memory manager,wherein the memory manager comprises an annotation API,
 - wherein the annotation API is configured to annotate within the process structure table the susceptibility of the program to buffer overflow attacks, and
 - wherein the memory manager is configured to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table.
4. The data processing system of claim 2 further comprising:
 - a process structure table in data communication with the memory manager,

wherein the memory manager comprises an annotation API,
wherein the annotation API is configured to annotate within the process structure table the susceptibility of the program to buffer overflow attacks, and
wherein the memory manager is configured to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table.

5. The data processing system of claim 1 further comprising:
a process structure table in data communication with the memory manager,
and
an annotation program in data communication with the process structure table,
wherein the annotation program is configured to annotate within the process structure table the susceptibility of the program to buffer overflow attacks, and
wherein the memory manager is configured to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table.
6. The data processing system of claim 3 wherein the program is configured to call the annotation API if the program is susceptible to buffer overflow attacks,
the memory manager is configured to determine susceptibility upon a request to allocate an additional page of RAM for the program.
7. The data processing system of claim 4 wherein the program is configured to call the annotation API if the program is susceptible to buffer overflow attacks,
the memory manager is configured to determine susceptibility upon a request to allocate an additional page of RAM for the program.
8. The data processing system of claim 5 wherein the program is configured to call the annotation program if the program is susceptible to buffer overflow attacks,
the memory manager is configured to determine susceptibility upon a request to allocate an additional page of RAM for the program.

9. A computer program product in a computer-readable medium adapted to prevent buffer overflow attacks comprising:

a memory manager code comprising a set of codes operable to direct a data processing system to manage a set of pages within a RAM of the data processing system and to permit a CPU of the data processing system to execute data on pages according to an execution flag on each of the set of pages;

an application program code comprising a set of codes operable to direct a data processing system to request the memory manager code to establish a program stack within at least one page the RAM; and

a susceptibility code comprising a set of codes operable to direct a data processing system to determine whether the application program code is susceptible to buffer overflow attacks, and, if so, set the execution flag for the program stack pages to deny CPU execution of data on the program stack pages.

10. The computer program product of claim 9 wherein the memory manager code further comprises a set of codes operable to direct a data processing system to deny CPU execution of data by triggering a hardware interrupt.

11. The computer program product of claim 9 further comprising:

a process structure table code comprising a set of codes operable to direct a data processing system to establish and maintain a process structure table code in data communication with the memory manager code and to annotate within the process structure table the susceptibility of the application program code to buffer overflow attacks,

wherein the memory manager code further comprises codes operable to direct a data processing system to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table.

12. The computer program product of claim 10 further comprising:
a process structure table code comprising a set of codes operable to direct a data processing system to establish and maintain a process structure table code in data communication with the memory manager code and to annotate within the process structure table the susceptibility of the application program code to buffer overflow attacks,
wherein the memory manager code further comprises codes operable to direct a data processing system to make the determination of susceptibility to buffer overflow attacks with reference to the process structure table.
13. The computer program product of claim 11 wherein the memory manager code comprises the process structure table code as an API.
14. The computer program product of claim 12 wherein the memory manager code comprises the process structure table code as an API.
15. The computer program product of claim 11 wherein the application program code further comprises a set of codes operable to direct a data processing system to call the process structure table code the application program code is susceptible to buffer overflow attacks, and
the memory manager code further comprises a set of codes operable to direct a data processing system to determine susceptibility upon receipt of a request to allocate an additional page of RAM for the application program code.
16. The computer program product of claim 14 wherein the application program code further comprises a set of codes operable to direct a data processing system to call the process structure table code the application program code is susceptible to buffer overflow attacks, and

the memory manager code further comprises a set of codes operable to direct a data processing system to determine susceptibility upon receipt of a request to allocate an additional page of RAM for the application program code.

17. The computer program product of claim 13 wherein the application program code further comprises a set of codes operable to direct a data processing system to call the process structure table code the application program code is susceptible to buffer overflow attacks, and

the memory manager code further comprises a set of codes operable to direct a data processing system to determine susceptibility upon receipt of a request to allocate an additional page of RAM for the application program code.

18. A method for handling buffer overflow attacks against an application program running on a data processing system, having a CPU and a RAM divided into pages, the method comprising the steps of:

designating an execution flag for each page of RAM allocated to a stack of the application program;

permitting CPU execution of data on pages of RAM according to the execution flag;

determining whether the application program is susceptible to buffer overflow attacks; and

if the application program is susceptible to buffer overflow attacks, setting the execution flag as to the pages of RAM allocated to the stack of the application program.

19. The method of claim 18 wherein step of permitting CPU execution comprises the steps of:

examining the execution flag on the page of RAM;

if the execution flag is set, triggering a hardware interrupt; and

otherwise executing the data on the page.

20. The method of claim 18 further comprising the steps of:
establishing a process structure table;
maintaining the process structure table by annotating within the process structure table the susceptibility of the application program to buffer overflow attacks,
wherein the step of determining susceptibility to buffer overflow attacks is made with reference to the process structure table.
21. The method of claim 19 further comprising the steps of:
establishing a process structure table;
maintaining the process structure table by annotating within the process structure table the susceptibility of the application program to buffer overflow attacks,
wherein the step of determining susceptibility to buffer overflow attacks is made with reference to the process structure table.
22. The method according to claim 20 wherein the step of maintaining the process structure table is done once at the beginning of execution of the application program,
and
the step of determining susceptibility is performed upon each receipt of a request to allocate an additional page of RAM for the application program code.
23. The method according to claim 21 wherein the step of maintaining the process structure table is done once at the beginning of execution of the application program,
and
the step of determining susceptibility is performed upon each receipt of a request to allocate an additional page of RAM for the application program code.

24. The method according to claim 22 wherein request to allocate an additional page of RAM is a request to allocate the additional page of RAM for the stack.

25. The method according to claim 23 wherein request to allocate an additional page of RAM is a request to allocate the additional page of RAM for the stack.

EVIDENCE APPENDIX

No evidence was submitted pursuant to §§1.130, 1.131, or 1.132 of 37 C.F.R. or of any other evidence entered by the Examiner and relied upon by Appellant in the Appeal.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings to the current proceeding.

Austin_1 452768v.1